Royaume du Maroc
Haut Commissariat au Plan
Institut National de Statistique
et d'Économie Appliquée
INSEA

Sujet:

Établissement d'un Pipeline de Données pour le Reporting Automatisé

Année académique: 2023-2024

Stage PFA

Réalisé par : EL RHIABI Soufiane - DSE
Encadré par : OULLOUK Fatima Ezzahra

Résumé

Ce projet s'inscrit dans une démarche de transformation digitale au sein de la Société Centrale de Réassurance (SCR), avec pour objectif la mise en place d'un pipeline de données automatisé et robuste. En intégrant des technologies modernes telles qu'Apache Airflow pour l'orchestration des tâches et dbt pour la transformation des données, ce pipeline permet une gestion fluide et contrôlée des processus ETL (Extract, Transform, Load).

La conception de ce pipeline repose également sur l'utilisation de PostgreSQL comme entrepôt de données et de Metabase pour la visualisation, offrant ainsi des capacités analytiques renforcées aux décideurs de la SCR. Great Expectations a été intégré pour garantir la qualité des données, assurant ainsi des analyses fiables et des rapports précis.

L'objectif final de ce projet est d'automatiser et d'optimiser le traitement des données afin de répondre aux exigences analytiques de la SCR et de soutenir les processus décisionnels de manière plus rapide et plus efficace. Ce pipeline pose les bases pour une infrastructure de données évolutive, capable de s'adapter aux futurs besoins de l'entreprise.

Remerciements

Je tiens à exprimer ma profonde gratitude envers tous ceux qui ont contribué de près ou de loin à la réussite de ce projet de stage.

En premier lieu, je remercie chaleureusement mes parents pour leur soutien inconditionnel, leurs encouragements constants, et la confiance qu'ils ont placée en moi tout au long de mon parcours académique et professionnel. Leur présence et leurs valeurs m'ont toujours guidé et motivé.

Je remercie également ma tante, chez qui j'ai résidé pendant toute la durée de mon stage. Son hospitalité, sa bienveillance, et son soutien quotidien ont créé un environnement propice à la concentration et à la réussite de mon projet.

Je souhaite exprimer ma reconnaissance envers mes superviseurs, Madame Fatima Zahra Oullouk et Monsieur Outhman Hijazi, pour leur accompagnement et leurs précieux conseils techniques. Leur expertise et leur disponibilité ont été essentiels à l'aboutissement de ce projet. Leurs retours constructifs m'ont permis de progresser et de renforcer mes compétences dans le domaine.

Enfin, je remercie toute l'équipe de la Société Centrale de Réassurance pour leur accueil et le cadre professionnel enrichissant qu'ils m'ont offert, me permettant ainsi d'apprendre et d'évoluer dans un environnement stimulant et bienveillant.

Table des matières

1 Introduction **7**

1 Contexte et Importance du Projet 7

2 Motivation du Projet 7

3 Objectifs de l'Étude 8

4 Structure du Rapport 8

5 Conclusion 9

2 Présentation de l'Organisme d'Accueil : Société Centrale de Réassurance (SCR) **10**

1 Introduction 10

2 Fondation et Évolution Historique 11

3 Positionnement et Reconnaissance Mondiale 11

4 Infrastructure et Développement Stratégique 12

5 Contributions au Marché Marocain et International 12

6 Engagement envers l'Innovation et la Transformation Digitale 13

7 Conclusion 14

3 Architecture et Technologies **15**

1 Vue d'Ensemble de l'Architecture 15

2 Technologies Utilisées 16

2.1 Docker et Conteneurisation 16

2.2 Orchestration des Tâches avec Apache Airflow 16

2.3 Modélisation et Transformation des Données avec dbt 17

2.4 Entrepôt de Données avec PostgreSQL 17

2.5 Validation de la Qualité des Données avec Great Expectations . . 18

2.6 Visualisation des Données avec Metabase 18

2.7 Python pour les Scripts et Automatisation 18

3 Flux de Données et Interactions entre Composants 19
4 Conclusion 19

4 Conception et Modélisation des Données 20
1 Contexte et Justification de la Conception 20
2 Identification des Processus Métiers 20
3 Modélisation des Données : De la 3NF au Modèle Dimensionnel 21
3.1 Transformation du Modèle Normalisé vers un Modèle Dimensionnel 21
3.2 Choix entre le Schéma en Étoile et le Schéma en Flocon 23
4 Identification des Tables de Faits et de Dimensions 24
4.1 Tables de Faits 24
4.2 Tables de Dimensions 24
5 Modélisation et Transformation des Données avec dbt 25
6 Conclusion 26

5 Implémentation du Pipeline de Données 27
1 Mise en Place de l'Infrastructure 27
1.1 Configuration de Docker et Docker Compose 27
2 Orchestration avec Apache Airflow 28
3 Modélisation et Transformations avec dbt 29
4 Stockage des Données dans PostgreSQL 30
5 Visualisation et Reporting avec Metabase 30
6 Défis Techniques et Solutions 31
7 Conclusion 31

6 Tests et Validation 32
1 Stratégie de Test 32
2 Tests de Qualité des Données avec dbt 32
3 Vérification de la Qualité des Données avec Great Expectations 33
4 Défis et Solutions Apportées 34
5 Conclusion 34

7 Conclusion 35
1 Résumé du Projet 35

2 Principaux Accomplissements 35
3 Défis et Enseignements 36
4 Perspectives d'Amélioration et Projets Futurs 36
5 Conclusion Générale 37

A Annexe : Code Source **38**

B Annexe B : Exemples de Code **39**
1 Exemple de Configuration Great Expectations 39
2 Exemple de Transformation avec dbt 42

Table des figures

1.1 Vue d'ensemble de la SCR et de ses valeurs 10
2.1 Siège historique de la SCR, symbole de sa fondation en 1960 11
3.1 Carte de la présence internationale de la SCR 12
5.1 Contributions stratégiques de la SCR au marché de la réassurance 13

1.1 Architecture globale du pipeline de données 15
2.1 Docker : Conteneurisation et Isolation des Environnements 16
2.2 Apache Airflow : Orchestration des Tâches ETL 16
2.3 Structure d'un DAG dans Apache Airflow 17
2.4 dbt : Modélisation et Transformation des Données 17
2.5 PostgreSQL : Entrepôt de Données 17
2.6 Great Expectations : Validation de la Qualité des Données 18
2.7 Metabase : Visualisation des Données 18
2.8 Python : Langage pour Scripts et Automatisation 18

1.1 Processus général de modélisation des données 20
2.1 Schéma source du processus de ventes dans *AdventureWorks* 21
3.1 Transformation du modèle 3NF vers un modèle dimensionnel 22
3.2 Schéma en étoile appliqué au processus de ventes 23

3.3 Exemple d'implémentation d'un schéma en flocon pour des hiérarchies complexes 24
4.1 Structure de la table de faits `fact_sales` 24
4.2 Table de dimension `dim_product` 25
5.1 DAG des transformations dbt pour le pipeline de données 26

1.1 Configuration des services via Docker Compose (emplacement réservé) . . 28
2.1 Structure d'un DAG typique dans Apache Airflow 28
3.1 Flux de travail de dbt pour la transformation des données 29
4.1 Schéma de la base de données PostgreSQL (emplacement réservé) 30
5.1 Exemple de tableau de bord interactif avec Metabase 30

3.1 Interface de Great Expectations pour les tests de qualité des données . . 33

Chapitre 1

Introduction

1 Contexte et Importance du Projet

Le secteur de la réassurance connaît une transformation rapide, stimulée par des innovations technologiques et des besoins accrus en efficacité opérationnelle. Selon un rapport de [source], le marché mondial de la réassurance a connu une croissance importante en raison de l'adoption accrue des technologies de transformation digitale. La Société Centrale de Réassurance (SCR), en tant qu'acteur clé sur le marché international, a intégré des initiatives de transformation digitale pour renforcer ses capacités analytiques et optimiser ses processus. Cette démarche est essentielle pour maintenir sa compétitivité et répondre aux exigences d'un marché en constante évolution, tout en s'alignant avec sa vision stratégique d'améliorer sa résilience opérationnelle et sa position de leader en Afrique.

2 Motivation du Projet

Mon stage au sein du département de transformation digitale de la SCR s'inscrit dans cette dynamique d'innovation et de modernisation des infrastructures IT. L'objectif principal était de concevoir et de déployer un pipeline de données moderne capable d'intégrer et de traiter de grands volumes de données, tout en garantissant des visualisations analytiques précises et pertinentes. Cette initiative répond aux besoins spécifiques de la SCR, tels que la production plus fréquente et plus automatisée de rapports analytiques. Pour cela, des technologies avancées telles que *Docker*, *Apache Airflow*, *dbt (Data Build*

Tool), et *PostgreSQL* ont été exploitées pour offrir une solution scalable et résiliente.

3 Objectifs de l'Étude

Le projet visait à :

— Mettre en place un entrepôt de données performant et fiable, capable de soutenir des analyses avancées de manière continue.

— Intégrer des processus ETL automatisés pour améliorer la gestion des données et minimiser les erreurs manuelles.

— Assurer des visualisations et des rapports dynamiques pour un support décisionnel efficace, permettant aux utilisateurs d'accéder à des informations actualisées et stratégiques de manière régulière.

Ces objectifs visent à répondre aux attentes croissantes en matière de transparence et de performance analytique au sein de l'entreprise, tout en renforçant la compétitivité de la SCR sur le marché international.

4 Structure du Rapport

Ce rapport se divise en plusieurs chapitres pour offrir une vue d'ensemble complète du projet :

— **Chapitre 2** : Présentation de l'organisme d'accueil, la SCR, ses activités, et son impact sur le marché, avec un focus sur ses défis et opportunités en matière de transformation digitale.

— **Chapitre 3** : Description de l'architecture et des technologies employées dans la mise en place du pipeline de données, incluant des illustrations sur la configuration de l'infrastructure technique.

— **Chapitre 4** : Détails sur la conception des modèles de données et les processus de modélisation, expliquant les choix de modélisation et leur impact sur les capacités d'analyse.

— **Chapitre 5** : Explication de l'implémentation des solutions techniques, y compris les défis rencontrés et les solutions mises en œuvre.

— **Chapitre 6** : Stratégies de tests et validation des résultats pour assurer la qualité des données et la robustesse du pipeline.

— **Chapitre 7** : Conclusion et perspectives pour les futurs projets de transformation digitale et recommandations pour l'amélioration continue.

5 Conclusion

L'introduction de ce rapport vise à contextualiser l'importance du projet et à présenter la structure qui guidera le lecteur à travers l'ensemble des aspects techniques et analytiques de la solution développée. Cette analyse mettra en lumière l'impact positif de la transformation digitale sur l'efficacité et la compétitivité de la SCR, démontrant comment l'intégration des technologies modernes peut renforcer ses capacités analytiques et sa résilience dans un marché dynamique.

Chapitre 2

Présentation de l'Organisme d'Accueil : Société Centrale de Réassurance (SCR)

1 Introduction

Avant de plonger dans notre étude, il est essentiel de présenter la Société Centrale de Réassurance (SCR), l'organisme d'accueil, afin de mieux comprendre son rôle, son impact sur le marché de la réassurance et sa pertinence dans le cadre de ce projet. Ce chapitre met en lumière les origines, l'évolution, le positionnement, ainsi que les contributions stratégiques de la SCR au Maroc et à l'international.

FIGURE 1.1 – Vue d'ensemble de la SCR et de ses valeurs

2 Fondation et Évolution Historique

Fondée en 1960 par une convention entre l'État Marocain et la Caisse de Dépôt et de Gestion (CDG), la SCR représente un pilier de la réassurance au Maroc. Avec la CDG comme actionnaire majoritaire à hauteur de 94% du capital, la SCR a évolué d'une réassurance conventionnelle vers une entité à portée mondiale.

FIGURE 2.1 – Siège historique de la SCR, symbole de sa fondation en 1960

3 Positionnement et Reconnaissance Mondiale

La SCR est reconnue pour sa solidité financière, validée par des notations telles que *B++ Good* par *AM BEST* et *AAA/stable* par *Fitch Rating*. Présente dans plus de 70 marchés à travers l'Afrique, le Moyen-Orient et l'Asie, elle collabore avec plus de 500 clients, consolidant ainsi sa position de leader dans le monde arabe et de troisième acteur majeur en Afrique en termes de primes souscrites.

FIGURE 3.1 – Carte de la présence internationale de la SCR

4 Infrastructure et Développement Stratégique

Le siège social, la *Tour Atlas*, construit en 1975, représente l'engagement durable de la SCR dans le secteur de la réassurance. La SCR a étendu ses activités à l'international par l'ouverture de bureaux stratégiques à Londres, Abidjan, Le Caire, Kigali et Johannesburg. Son intégration dans *Casablanca Finance City* et la création d'un centre de secours (*backup*) témoignent de sa stratégie de développement et de résilience.

5 Contributions au Marché Marocain et International

La SCR joue un rôle crucial dans la régulation et la stabilité du marché marocain de la réassurance. Elle contribue activement par :

— **Régulation du Marché** : En tant qu'acteur clé, la SCR garantit la transparence

et la stabilité des pratiques de réassurance.

— **Sécurité face aux Fluctuations Internationales** : Elle protège les compagnies d'assurance marocaines contre les risques systémiques globaux.

— **Mobilisation de l'Épargne Nationale** : En maintenant des primes d'assurance viables, la SCR stimule l'économie nationale en favorisant la mobilisation de l'épargne.

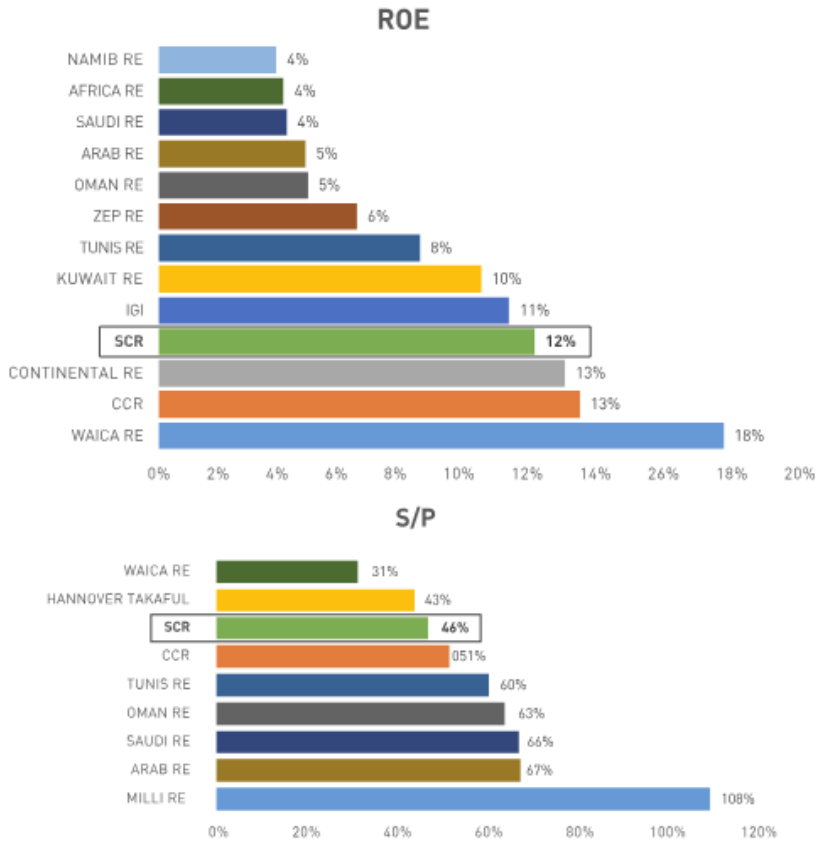

FIGURE 5.1 – Contributions stratégiques de la SCR au marché de la réassurance

6 Engagement envers l'Innovation et la Transformation Digitale

Pour rester compétitive, la SCR adopte des technologies modernes et investit dans la transformation digitale. Cette vision novatrice est essentielle pour améliorer ses processus opérationnels, répondre aux attentes des clients et se préparer aux défis futurs de l'industrie de la réassurance. Les initiatives de transformation digitale de la SCR jouent un rôle central dans la stratégie de l'entreprise pour maintenir une efficacité et une réactivité

optimales.

7 Conclusion

Cette présentation de la SCR permet de comprendre l'importance de ses activités et de son positionnement stratégique, non seulement pour le marché marocain, mais aussi au niveau international. Ces aspects jouent un rôle crucial dans l'analyse des initiatives technologiques et des projets de transformation digitale abordés dans les chapitres suivants.

Chapitre 3

Architecture et Technologies

1 Vue d'Ensemble de l'Architecture

L'architecture de ce projet repose sur un pipeline de données entièrement automatisé et conteneurisé. Conçue pour être scalable et performante, elle couvre l'ensemble du cycle de vie des données : collecte, stockage, transformation, et visualisation. Cette architecture modulaire et flexible est composée de plusieurs composants interconnectés, chacun jouant un rôle crucial dans le flux de données.

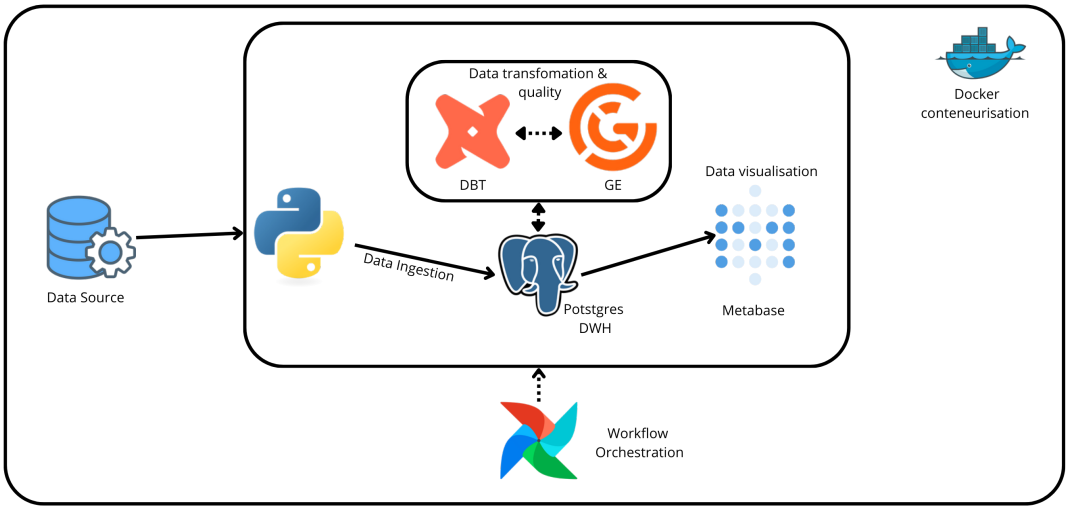

FIGURE 1.1 – Architecture globale du pipeline de données

2 Technologies Utilisées

Le pipeline s'appuie sur des technologies modernes pour garantir robustesse, maintenance simplifiée, et scalabilité. Voici les principaux outils utilisés :

2.1 Docker et Conteneurisation

FIGURE 2.1 – Docker : Conteneurisation et Isolation des Environnements

Docker a été utilisé pour conteneuriser tous les services du pipeline, assurant l'isolation des environnements et simplifiant le déploiement. La conteneurisation via Docker Compose garantit une gestion des dépendances entre les services et permet une portabilité accrue.

2.2 Orchestration des Tâches avec Apache Airflow

FIGURE 2.2 – Apache Airflow : Orchestration des Tâches ETL

Apache Airflow orchestre les tâches ETL du pipeline. À travers des DAGs (Directed Acyclic Graphs), chaque étape du flux de données est planifiée et contrôlée pour un suivi précis et une gestion automatisée.

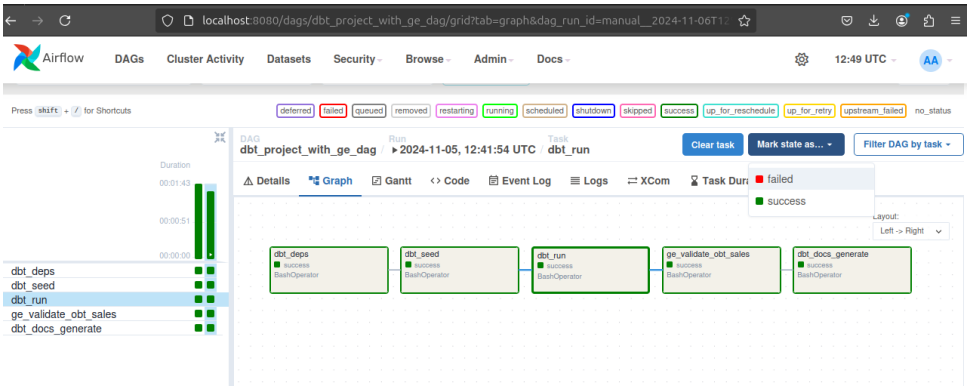

FIGURE 2.3 – Structure d'un DAG dans Apache Airflow

2.3 Modélisation et Transformation des Données avec dbt

FIGURE 2.4 – dbt : Modélisation et Transformation des Données

dbt (Data Build Tool) gère les transformations SQL du pipeline. Son système de modèles assure la transparence et la traçabilité des transformations. En utilisant dbt, les modèles de données peuvent être versionnés et documentés pour faciliter leur gestion.

2.4 Entrepôt de Données avec PostgreSQL

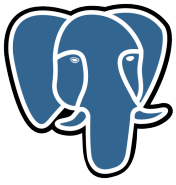

FIGURE 2.5 – PostgreSQL : Entrepôt de Données

PostgreSQL sert de base de données principale pour stocker les données transformées. Sa structure optimise le stockage et l'accès aux données analytiques. Dans cette architecture, PostgreSQL agit comme l'entrepôt central pour les données prêtes à l'analyse.

2.5 Validation de la Qualité des Données avec Great Expectations

FIGURE 2.6 – Great Expectations : Validation de la Qualité des Données

Great Expectations a été intégré pour assurer la qualité des données. Cet outil vérifie les valeurs, les schémas et les contraintes définies sur les données transformées, garantissant ainsi leur fiabilité avant les analyses finales.

2.6 Visualisation des Données avec Metabase

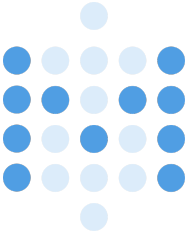

FIGURE 2.7 – Metabase : Visualisation des Données

Metabase est utilisé pour les visualisations. En se connectant à PostgreSQL, il permet de créer des tableaux de bord interactifs et des rapports, fournissant aux utilisateurs une vue en temps réel sur les indicateurs de performance clés (KPI) pour une prise de décision basée sur des données fiables.

2.7 Python pour les Scripts et Automatisation

FIGURE 2.8 – Python : Langage pour Scripts et Automatisation

Python a été utilisé pour développer des scripts personnalisés dans le pipeline. Ce langage a permis de créer des fonctions spécifiques de transformation et d'automatiser des tâches telles que l'intégration avec d'autres services et la préparation des données.

3 Flux de Données et Interactions entre Composants

Le pipeline de données suit un flux organisé où chaque composant joue un rôle distinct :

— **Ingestion** : Les données sont récupérées et chargées dans PostgreSQL.

— **Orchestration** : Airflow gère et suit les tâches de transformation, garantissant le bon déroulement du pipeline.

— **Transformation** : dbt applique les transformations SQL sur les données brutes pour les préparer à l'analyse.

— **Qualité** : Great Expectations valide la qualité des données selon des règles définies.

— **Stockage** : PostgreSQL stocke les données traitées et prêtes pour les requêtes analytiques.

— **Visualisation** : Metabase extrait les données depuis PostgreSQL et les présente sous forme de tableaux de bord interactifs.

4 Conclusion

Cette architecture tire parti de technologies conteneurisées pour répondre aux exigences de la Société Centrale de Réassurance en termes d'analyses de données. La combinaison de Docker, Airflow, dbt, PostgreSQL, Metabase, et Great Expectations offre une solution intégrée, automatisée et scalable pour une analyse de données fiable et en temps réel.

Chapitre 4

Conception et Modélisation des Données

1 Contexte et Justification de la Conception

Pour la réalisation de ce projet, j'ai utilisé la base de données *AdventureWorks*, en raison de contraintes d'accès aux données internes de la Société Centrale de Réassurance (SCR). L'utilisation de cette base de données publique permet de démontrer et de valider des concepts de modélisation dimensionnelle dans un contexte pertinent. La méthodologie adoptée suit une approche de modélisation en étoile, particulièrement adaptée aux besoins d'analyse et de reporting.

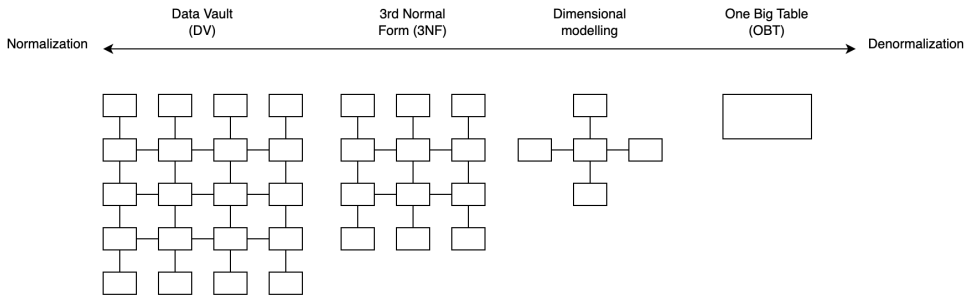

FIGURE 1.1 – Processus général de modélisation des données

2 Identification des Processus Métiers

Le processus métier modélisé est le **processus de ventes**, aligné avec les objectifs analytiques de la SCR. Cette modélisation offre une granularité détaillée des revenus par produit, client, statut de commande et localisation géographique, permettant des analyses approfondies et une prise de décision éclairée.

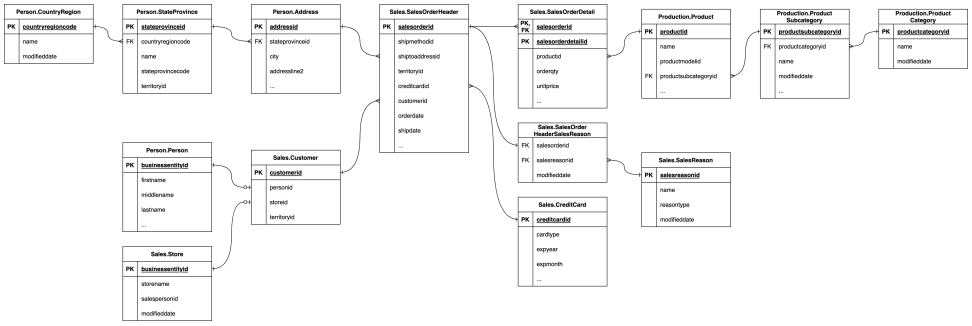

FIGURE 2.1 – Schéma source du processus de ventes dans *AdventureWorks*

3 Modélisation des Données : De la 3NF au Modèle Dimensionnel

Les données ont été transformées d'un modèle normalisé (3NF) en un modèle dimensionnel pour simplifier les requêtes analytiques. Le modèle en étoile a été privilégié pour une clarté et une flexibilité accrues dans les jointures nécessaires aux analyses.

3.1 Transformation du Modèle Normalisé vers un Modèle Dimensionnel

Le modèle relationnel de *AdventureWorks* en 3NF a été analysé pour identifier les tables de faits et de dimensions potentielles, assurant une structure adaptée aux analyses.

FIGURE 3.1 – Transformation du modèle 3NF vers un modèle dimensionnel

3.2 Choix entre le Schéma en Étoile et le Schéma en Flocon

Le schéma en étoile, avec une table de faits centrale reliée à plusieurs tables de dimensions, a été choisi pour faciliter la navigation et simplifier les analyses. Dans certains cas, un schéma en flocon a été utilisé pour répondre aux besoins de hiérarchies complexes.

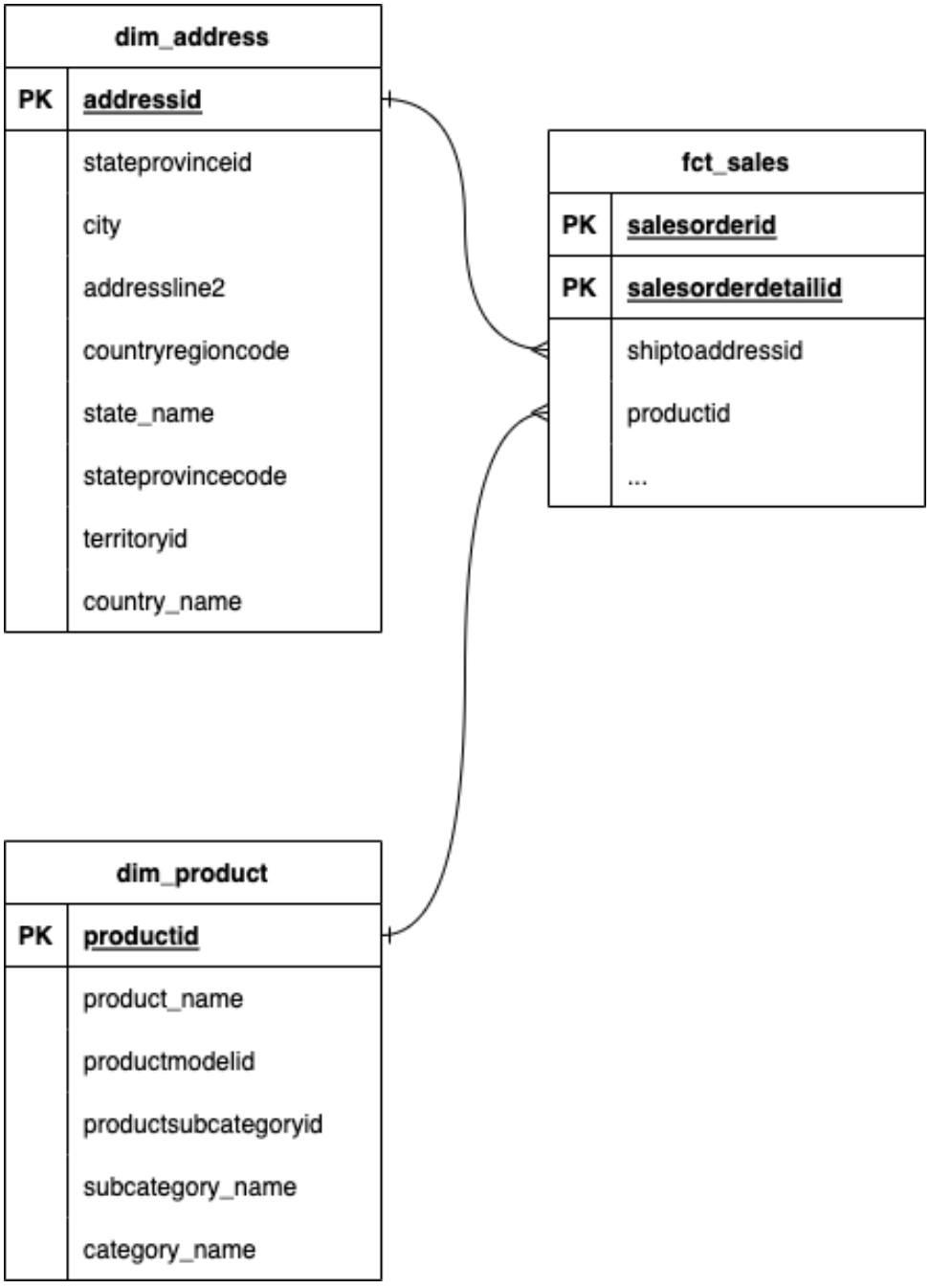

FIGURE 3.2 – Schéma en étoile appliqué au processus de ventes

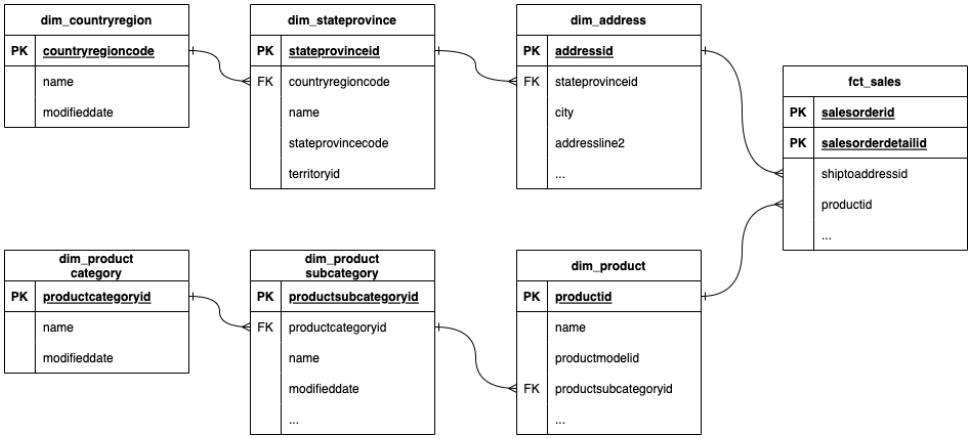

FIGURE 3.3 – Exemple d'implémentation d'un schéma en flocon pour des hiérarchies complexes

4 Identification des Tables de Faits et de Dimensions

Pour le processus de ventes, plusieurs tables ont été identifiées comme sources pour les tables de faits et de dimensions, fournissant une vue complète et cohérente des transactions et des attributs descriptifs associés.

4.1 Tables de Faits

La table de faits principale, `fact_sales`, capture les informations transactionnelles des ventes, incluant le montant, la quantité et les remises.

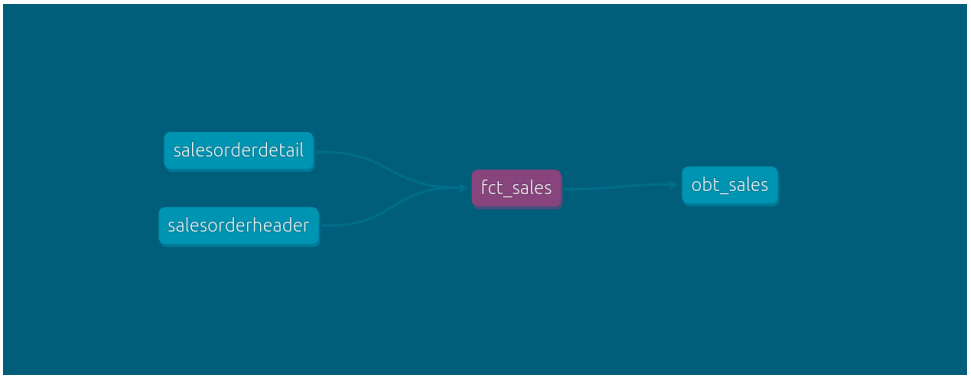

FIGURE 4.1 – Structure de la table de faits `fact_sales`

4.2 Tables de Dimensions

Les tables de dimensions enrichissent la table de faits avec des informations descriptives, permettant des analyses en profondeur : - `dim_product` : contient les détails

des produits, y compris la hiérarchie des catégories. - `dim_customer` : informations sur les clients, leurs segments et leur localisation. - `dim_address` : localisation géographique pour des analyses régionales. - `dim_order_status`, `dim_credit_card` et `dim_date` : capturent respectivement le statut des commandes, les informations de paiement et les données temporelles.

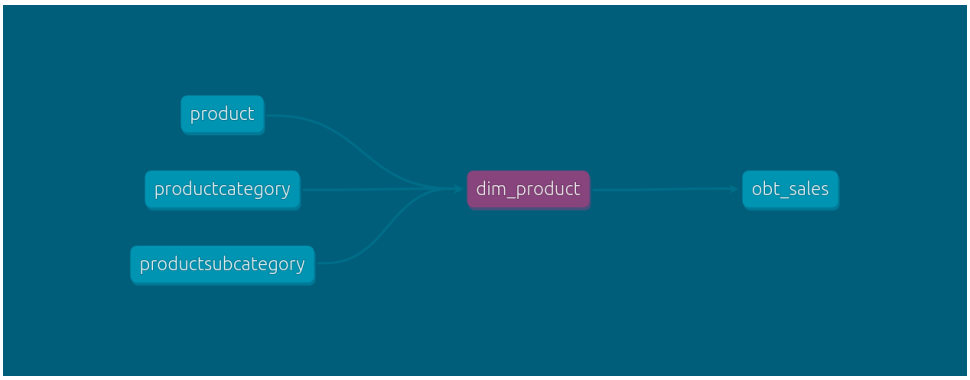

FIGURE 4.2 – Table de dimension `dim_product`

5 Modélisation et Transformation des Données avec dbt

Chaque transformation SQL a été gérée dans dbt, permettant une modélisation structurée et documentée des données. dbt permet également la visualisation de l'ordre des transformations via son graphe des dépendances.

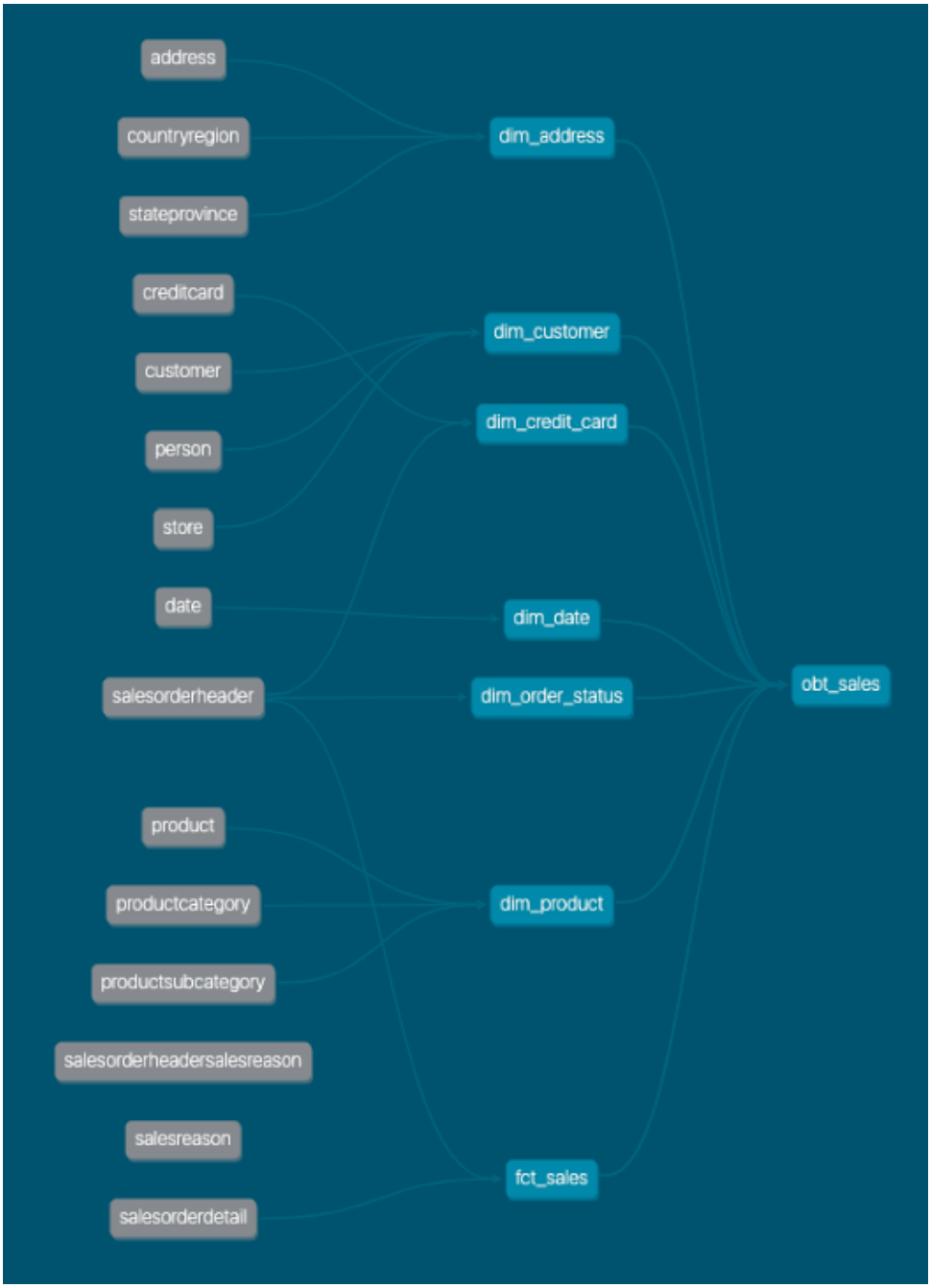

FIGURE 5.1 – DAG des transformations dbt pour le pipeline de données

6 Conclusion

La conception du data warehouse, structurée selon un modèle dimensionnel en étoile, fournit une base fiable pour les analyses de la SCR. Grâce à dbt, la gestion des transformations et la documentation des modèles sont assurées de manière efficace, offrant ainsi une transparence et une traçabilité précieuses.

Chapitre 5

Implémentation du Pipeline de Données

1 Mise en Place de l'Infrastructure

L'infrastructure du pipeline de données a été construite en utilisant une architecture conteneurisée avec Docker, permettant de créer un environnement stable, reproductible, et adapté aux différentes étapes de traitement des données. Cette approche simplifie le déploiement, assure la compatibilité entre composants, et facilite la maintenance.

1.1 Configuration de Docker et Docker Compose

Le fichier `docker-compose.yml` gère l'orchestration des principaux services, notamment Apache Airflow, dbt, PostgreSQL et Metabase. En coordonnant les conteneurs, il assure une intégration fluide entre chaque composant.

Listing 5.1 – Extrait du fichier docker-compose.yml

```
version: '3.8'
services:
  postgres:
    image: postgres:latest
    environment:
      POSTGRES_USER: user
      POSTGRES_PASSWORD: password
    ports:
```

```
    - "5432:5432"

  airflow:
    image: apache/airflow:latest
    environment:
      - AIRFLOW__CORE__LOAD_EXAMPLES=False
    ports:
      - "8080:8080"
```

```
CONTAINER ID   IMAGE                          COMMAND                  CREATED         STATUS                   PORTS
                                                                                       NAMES
9bfec63508da   airflow-sql-airflow-scheduler  "/usr/bin/dumb-init …"   5 minutes ago   Up 4 minutes (healthy)   8080/tcp
                                                                                       airflow-sql-airflow-scheduler-1
8e2b90e01725   airflow-sql-airflow-worker     "/usr/bin/dumb-init …"   5 minutes ago   Up 4 minutes (healthy)   8080/tcp
                                                                                       airflow-sql-airflow-worker-1
01a376e54cd1   airflow-sql-airflow-webserver  "/usr/bin/dumb-init …"   5 minutes ago   Up 4 minutes (healthy)   0.0.0.0:8001->8001/tcp, :::80
01->8001/tcp, 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:8888->8888/tcp, :::8888->8888/tcp     airflow-sql-airflow-webserver-1
d46298dad7f6   airflow-sql-airflow-triggerer  "/usr/bin/dumb-init …"   5 minutes ago   Up 4 minutes (healthy)   8080/tcp
                                                                                       airflow-sql-airflow-triggerer-1
5685f3c00a70   dpage/pgadmin4                 "/entrypoint.sh"         5 minutes ago   Up 5 minutes             443/tcp, 0.0.0.0:5050->80/tcp
, [::]:5050->80/tcp                                                                    pgadmin
45a3b327a3b1   metabase/metabase              "/app/run_metabase.sh"   5 minutes ago   Up 5 minutes             0.0.0.0:3000->3000/tcp, :::30
00->3000/tcp                                                                           metabase
43cc2dd9eda2   postgres:13                    "docker-entrypoint.s…"   5 minutes ago   Up 5 minutes (healthy)   5432/tcp
                                                                                       airflow-sql-postgres-1
670c920707a9   redis:7.2-bookworm             "docker-entrypoint.s…"   5 minutes ago   Up 5 minutes (healthy)   6379/tcp
                                                                                       airflow-sql-redis-1
```

FIGURE 1.1 – Configuration des services via Docker Compose (emplacement réservé)

2 Orchestration avec Apache Airflow

Apache Airflow a été utilisé pour orchestrer les tâches ETL (Extract, Transform, Load) du pipeline. Les DAGs (Directed Acyclic Graphs) définissent les étapes d'extraction, transformation et chargement des données. Ce système offre une gestion automatisée, un suivi en temps réel des tâches, et un contrôle de l'échec pour garantir un flux de travail fiable.

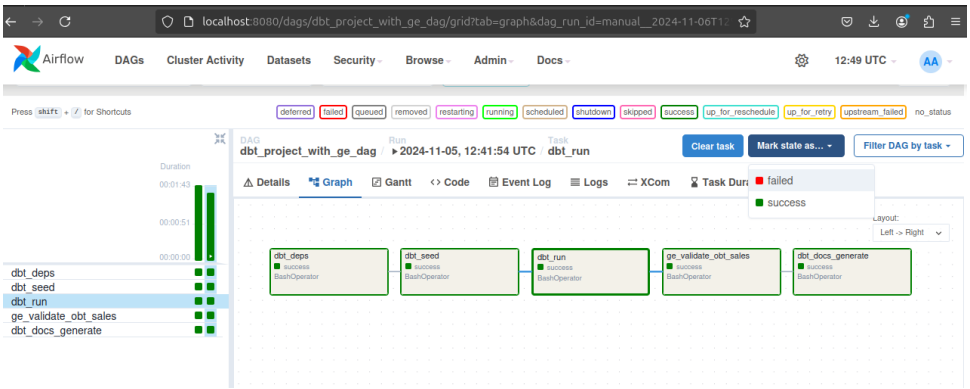

FIGURE 2.1 – Structure d'un DAG typique dans Apache Airflow

Le code complet des DAGs est fourni dans l'appendice (voir Annexe A).

3 Modélisation et Transformations avec dbt

Les transformations de données sont gérées avec dbt, facilitant la création de modèles SQL modulaires et documentés. Grâce à sa capacité de versionnement et de traçabilité, dbt permet un développement structuré et maintenable des transformations de données. Le flux de travail dbt est illustré dans la figure ci-dessous.

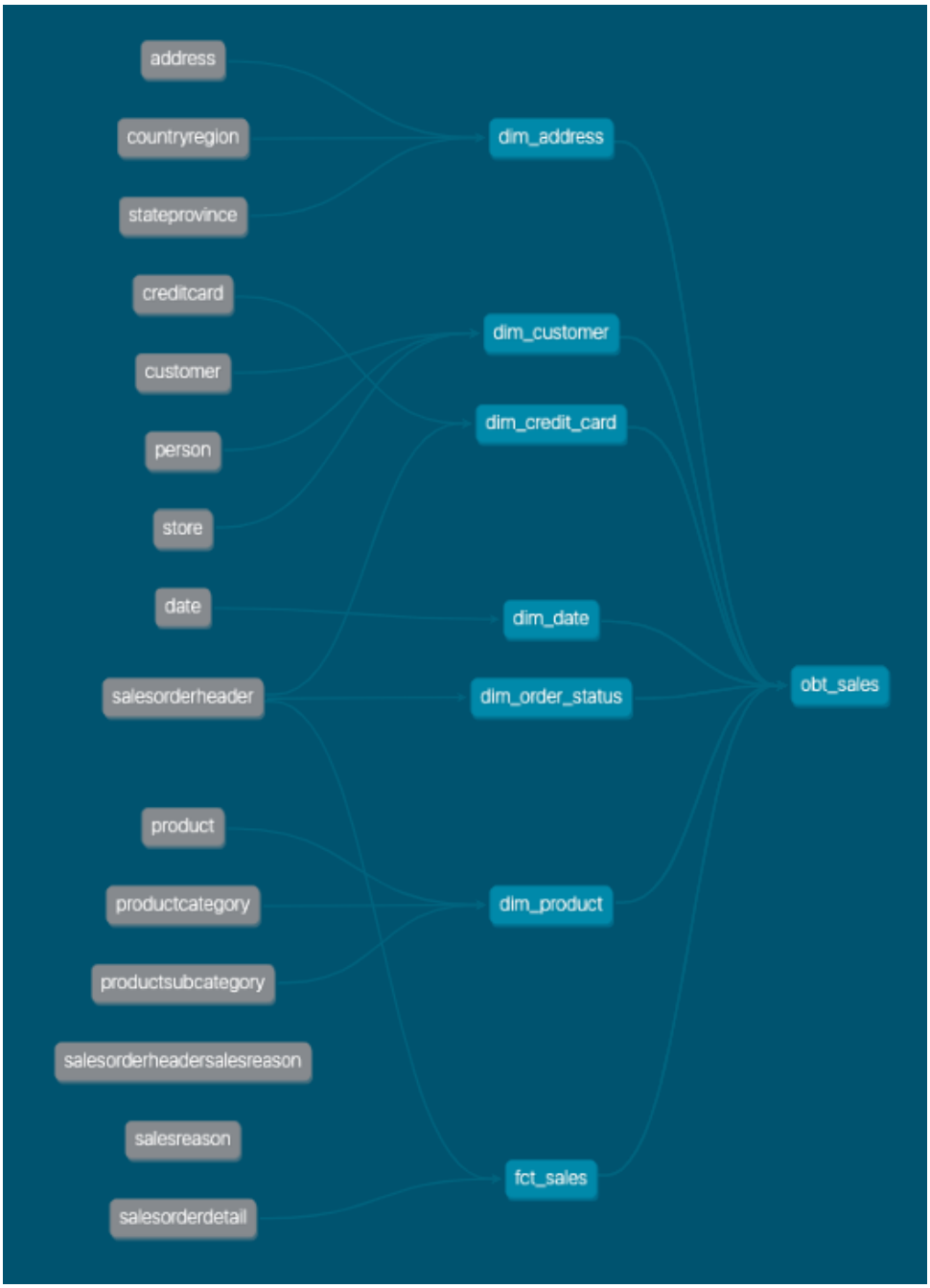

FIGURE 3.1 – Flux de travail de dbt pour la transformation des données

4 Stockage des Données dans PostgreSQL

PostgreSQL est configuré pour fonctionner en tant qu'entrepôt de données, permettant de stocker les résultats des transformations de manière centralisée et sécurisée. Les tables ont été optimisées pour les requêtes analytiques, avec des index et des structures de données ajustés pour un accès efficace aux informations. Cette configuration garantit une persistance des données et permet des analyses rapides.

FIGURE 4.1 – Schéma de la base de données PostgreSQL (emplacement réservé)

5 Visualisation et Reporting avec Metabase

Metabase est utilisé pour la visualisation des données. Sa connexion directe à PostgreSQL permet la création de tableaux de bord interactifs, qui facilitent le suivi des indicateurs clés de performance (KPI) et favorisent la prise de décision basée sur des analyses visuelles. Metabase offre une interface intuitive, adaptée aux besoins analytiques de la SCR.

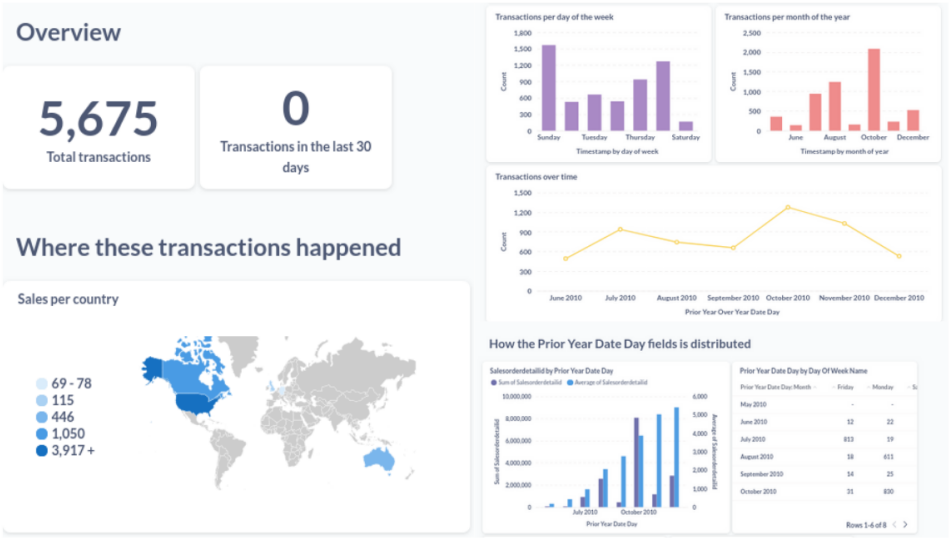

FIGURE 5.1 – Exemple de tableau de bord interactif avec Metabase

6 Défis Techniques et Solutions

Durant l'implémentation, plusieurs défis ont été rencontrés, nécessitant des ajustements techniques :

— **Optimisation des DAGs dans Airflow** : L'ordonnancement des tâches a été configuré pour une gestion efficace des dépendances et un contrôle des échecs de tâches.

— **Dépendances entre services Docker** : Des scripts d'initialisation ont été ajoutés pour garantir que chaque service était opérationnel avant de lancer les processus dépendants.

— **Configuration de PostgreSQL** : Les indices et structures de tables ont été optimisés pour gérer les volumes de données et faciliter les analyses.

7 Conclusion

L'implémentation de ce pipeline de données a permis de mettre en place une infrastructure automatisée et adaptable, répondant aux besoins d'analyse de la SCR. Les choix technologiques, incluant Docker, Airflow, dbt, PostgreSQL, et Metabase, démontrent la pertinence d'une architecture modulaire et évolutive pour la gestion de données complexes. Ce pipeline fournit une base solide pour les futurs projets de transformation digitale et peut être étendu en fonction des exigences analytiques futures.

Chapitre 6

Tests et Validation

1 Stratégie de Test

Pour garantir la qualité et la fiabilité du pipeline de données, une stratégie de test rigoureuse axée sur la validation des données a été mise en place. L'objectif principal de cette stratégie est de s'assurer que les données traitées et stockées répondent aux critères de qualité définis, notamment en termes de cohérence, d'exactitude et d'absence d'anomalies.

2 Tests de Qualité des Données avec dbt

dbt a été utilisé pour automatiser la validation des données tout au long du pipeline. En intégrant des tests de qualité directement dans les transformations SQL, dbt permet de vérifier des aspects clés tels que :

— **Non-nullité** : Vérification que des colonnes critiques ne contiennent pas de valeurs nulles, garantissant l'intégrité des données.

— **Conformité des types de données** : Vérification que les types de données correspondent aux spécifications, ce qui est crucial pour l'exactitude des calculs et des agrégations.

— **Unicité des clés primaires** : Assurance que les colonnes identifiées comme clés primaires ne contiennent pas de doublons, évitant ainsi les incohérences dans les analyses.

Voici un exemple de test de validation réalisé avec dbt pour s'assurer qu'une colonne critique ne contient pas de valeurs nulles :

Listing 6.1 – Exemple de test de validation avec dbt

```
-- Test de validation de la non-nullit  d'une colonne
SELECT *
FROM {{ ref('table_model') }}
WHERE column_name IS NULL;
```

3 Vérification de la Qualité des Données avec Great Expectations

Great Expectations a été utilisé pour des tests de qualité des données plus avancés, offrant une interface permettant de définir et de surveiller des suites de tests automatisés sur les données. Cette approche permet de vérifier de manière détaillée que les données respectent des attentes spécifiques, telles que la distribution des valeurs, la plage acceptable, et la cohérence globale des données.

L'interface de Great Expectations facilite la configuration de ces tests et la gestion des rapports de qualité, fournissant une traçabilité essentielle pour les validations.

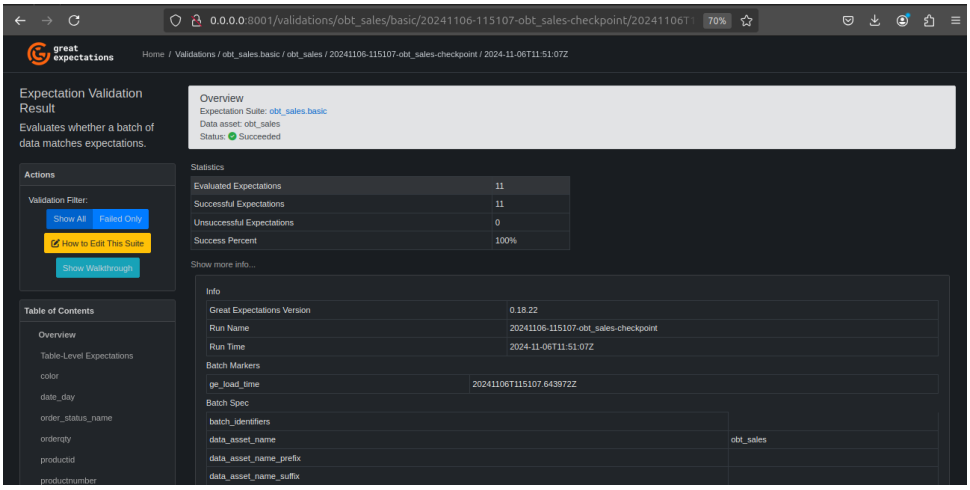

FIGURE 3.1 – Interface de Great Expectations pour les tests de qualité des données

4 Défis et Solutions Apportées

Au cours de la phase de tests de qualité, quelques défis ont été rencontrés et résolus pour améliorer la robustesse des validations :

— **Optimisation des Tests SQL dans dbt** : Pour s'assurer que les tests de qualité ne ralentissent pas le pipeline, les requêtes SQL dans dbt ont été optimisées afin de réduire le temps de traitement sans compromettre l'exactitude.

— **Alertes de Qualité en Temps Réel** : Great Expectations a permis de mettre en place des notifications pour signaler les échecs de tests de qualité, garantissant une détection rapide des anomalies.

— **Gestion des Ressources Docker** : Des paramètres de ressources spécifiques ont été configurés pour chaque conteneur, assurant l'efficacité de l'exécution des tests dbt et Great Expectations.

5 Conclusion

La stratégie de test axée sur la qualité des données a permis de garantir la fiabilité des informations utilisées dans les analyses de la SCR. Les validations réalisées avec dbt et les contrôles via Great Expectations ont assuré que les données respectent les standards de qualité requis, permettant ainsi des décisions basées sur des données fiables et précises. Cette approche contribue à renforcer la confiance dans les analyses produites par le pipeline de données et fournit une base solide pour les futurs besoins analytiques de la SCR.

Chapitre 7

Conclusion

1 Résumé du Projet

Ce projet a permis de concevoir, développer et valider un pipeline de données automatisé pour la Société Centrale de Réassurance (SCR), en utilisant des technologies modernes telles qu'*Apache Airflow*, *dbt (Data Build Tool)*, *PostgreSQL*, et *Metabase*. L'approche adoptée a démontré la viabilité de l'automatisation des processus ETL, permettant ainsi une gestion plus structurée et précise des flux de données, avec des capacités analytiques renforcées.

2 Principaux Accomplissements

Plusieurs objectifs clés ont été atteints :

— **Automatisation des Processus ETL** : Grâce à Airflow, l'extraction, la transformation et le chargement des données se déroulent désormais de manière automatique, réduisant les risques d'erreurs manuelles et améliorant la disponibilité des données pour l'analyse.

— **Modélisation Dimensionnelle avec dbt** : La mise en œuvre de modèles en étoile dans dbt a permis une organisation structurée des données, facilitant les analyses approfondies et optimisant la clarté des rapports.

— **Amélioration de la Qualité des Données** : Les tests et validations intégrés ont assuré une cohérence et une fiabilité des données, répondant aux standards de qualité requis par la SCR.

— **Visualisation Dynamique des Données** : Metabase a été intégré pour créer des tableaux de bord interactifs, offrant ainsi aux décideurs un accès direct et en temps réel aux indicateurs de performance.

3 Défis et Enseignements

Au cours de l'implémentation, plusieurs défis techniques et méthodologiques ont été relevés :

— **Gestion des Dépendances des Tâches** : La complexité des dépendances dans Airflow a nécessité une configuration fine des DAGs et une optimisation du *scheduler* pour garantir un flux de données fluide.

— **Optimisation des Performances** : La gestion de grands volumes de données a mis en évidence la nécessité d'optimiser les requêtes SQL et de configurer les ressources des conteneurs Docker pour assurer des temps de traitement acceptables.

— **Validation et Qualité des Données** : Assurer la qualité des données via des tests de validation automatisés avec dbt et Great Expectations a permis de détecter et de corriger des incohérences, un aspect essentiel pour maintenir des standards élevés de précision.

Ces défis ont permis de développer une expertise technique approfondie et d'identifier des pratiques optimales pour la gestion de pipelines de données, particulièrement en contexte de réassurance.

4 Perspectives d'Amélioration et Projets Futurs

Les résultats obtenus ouvrent plusieurs perspectives pour de futurs projets et améliorations potentielles :

— **Scalabilité et Performance** : Envisager l'utilisation de technologies distribuées, telles que *Apache Spark* ou *Dask*, pour gérer des volumes de données plus importants et mieux adapter le pipeline à des charges de données croissantes.

— **Automatisation Avancée des Tests de Données** : Mettre en place des tests plus sophistiqués de qualité de données avec *Great Expectations*, afin d'intégrer des validations supplémentaires et de renforcer la surveillance continue de la qualité.

— **Extension du Modèle de Données** : Explorer l'ajout de nouveaux processus métiers dans le data warehouse, pour élargir les capacités analytiques de la SCR et inclure davantage de départements dans le processus de reporting et de visualisation des données.

5 Conclusion Générale

En conclusion, ce projet a démontré comment un pipeline de données moderne peut transformer l'accès aux informations et l'efficacité analytique pour la Société Centrale de Réassurance. L'implémentation de ce pipeline a permis à la SCR de disposer d'une infrastructure évolutive et automatisée, alignée sur ses objectifs de transformation digitale. À travers une intégration cohérente des technologies et une validation rigoureuse de la qualité des données, ce projet constitue une base solide pour les futures initiatives de la SCR, en soutien à une prise de décision stratégique, fiable et basée sur des données précises.

Annexe A

Annexe : Code Source

Le code source complet de ce projet, incluant les configurations d'Airflow, dbt, Docker, et PostgreSQL, ainsi que les scripts SQL et Python utilisés, est disponible dans le dépôt GitHub suivant :

— https://github.com/SELRHIABI/DataPipeline

Annexe B

Annexe B : Exemples de Code

1 Exemple de Configuration Great Expectations

Le snippet suivant illustre une suite d'expectations configurée dans Great Expectations pour valider la qualité des données. Ce code vérifie plusieurs aspects tels que l'unicité des valeurs, l'absence de valeurs nulles dans des colonnes critiques, le type de données et la conformité à des formats spécifiques.

Listing B.1 – Configuration Great Expectations pour la validation de données

```
{
  "data_asset_type": null,
  "expectation_suite_name": "obt_sales.basic",
  "expectations": [
    {
      "expectation_type": "expect_table_row_count_to_be_between",
      "kwargs": {
        "min_value": 1
      }
    },
    {
      "expectation_type": "expect_column_values_to_be_unique",
      "kwargs": {
        "column": "sales_key"
```

```
      }
    },
    {
      "expectation_type": "expect_column_values_to_not_be_null",
      "kwargs": {
        "column": "salesorderid"
      }
    },
    {
      "expectation_type": "expect_column_values_to_not_be_null",
      "kwargs": {
        "column": "salesorderdetailid"
      }
    },
    {
      "expectation_type": "expect_column_values_to_not_be_null",
      "kwargs": {
        "column": "unitprice"
      }
    },
    {
      "expectation_type": "expect_column_values_to_be_of_type",
      "kwargs": {
        "column": "orderqty",
        "type_": "SMALLINT"
      }
    },
    {
      "expectation_type": "expect_column_values_to_not_be_null",
      "kwargs": {
        "column": "productid"
      }
```

```
    },
    {
      "expectation_type": "expect_column_values_to_be_in_set",
      "kwargs": {
        "column": "color",
        "value_set": ["Black", "Silver", "White", "Red", null]
      }
    },
    {
      "expectation_type": "expect_column_values_to_match_regex",
      "kwargs": {
        "column": "productnumber",
        "regex": "^[A-Z]{2}-[A-Z0-9]{3,4}(-[A-Z0-9]{1,2})?$"
      }
    },
    {
      "expectation_type": "expect_column_values_to_be_in_set",
      "kwargs": {
        "column": "order_status_name",
        "value_set": ["shipped", "pending", "canceled"]
      }
    },
    {
      "expectation_type": "expect_column_values_to_not_be_null",
      "kwargs": {
        "column": "date_day"
      }
    }
  ],
  "ge_cloud_id": null,
  "meta": {
    "great_expectations_version": "0.18.22"
```

```
  }
}
```

2 Exemple de Transformation avec dbt

Voici un exemple de code dbt pour une transformation SQL. Ce code combine plusieurs tables de faits et de dimensions pour construire un modèle de données complet, en utilisant des jointures et la fonction `dbt_utils.star` pour sélectionner toutes les colonnes sauf certaines clés étrangères.

Listing B.2 – Transformation SQL avec dbt

```
with f_sales as (
    select * from {{ ref('fct_sales') }}
),

d_customer as (
    select * from {{ ref('dim_customer') }}
),

d_credit_card as (
    select * from {{ ref('dim_credit_card') }}
),

d_address as (
    select * from {{ ref('dim_address') }}
),

d_order_status as (
    select * from {{ ref('dim_order_status') }}
),

d_product as (
    select * from {{ ref('dim_product') }}
```

```
),

d_date as (
    select * from {{ ref('dim_date') }}
)

select
    {{ dbt_utils.star(from=ref('fct_sales'), relation_alias='f_sales', exce
        "product_key", "customer_key", "creditcard_key", "ship_address_key"
    ]) }},
    {{ dbt_utils.star(from=ref('dim_product'), relation_alias='d_product',
    {{ dbt_utils.star(from=ref('dim_customer'), relation_alias='d_customer
    {{ dbt_utils.star(from=ref('dim_credit_card'), relation_alias='d_credit
    {{ dbt_utils.star(from=ref('dim_address'), relation_alias='d_address',
    {{ dbt_utils.star(from=ref('dim_order_status'), relation_alias='d_orde
    {{ dbt_utils.star(from=ref('dim_date'), relation_alias='d_date', except
from f_sales
left join d_product on f_sales.product_key = d_product.product_key
left join d_customer on f_sales.customer_key = d_customer.customer_key
left join d_credit_card on f_sales.creditcard_key = d_credit_card.creditcard
left join d_address on f_sales.ship_address_key = d_address.address_key
left join d_order_status on f_sales.order_status_key = d_order_status.order
left join d_date on f_sales.order_date_key = d_date.date_key
```